



PAOLO CAMAGNI
RICCARDO NIKOLASSY

INFORMATICA IN C++ E JAVA

Per il secondo biennio degli Istituti Tecnici
Tecnologici, articolazione Informatica

La programmazione imperativa in C, C++ e Java
La programmazione ad oggetti in C++ e Java
Le interfacce grafiche in C# e Java
La programmazione del web in HTML e Javascript



**HOEPLI
TECNICA
PER LA SCUOLA**

Edizione **OPENSCHOOL**

1	LIBRODITESTO
2	E-BOOK+
3	RISORSEONLINE
4	PIATTAFORMA



HOEPLI

Informatica in C++ e Java

PAOLO CAMAGNI RICCARDO NIKOLASSY

Informatica in C++ e Java

La programmazione imperativa in C, C++ e Java

La programmazione ad oggetti in C++ e Java

Le interfacce grafiche in C# e Java

La programmazione del web in HTML e Javascript



EDITORE ULRICO HOEPLI MILANO

Copyright © Ulrico Hoepli Editore S.p.A. 2019

Via Hoepli 5, 20121 Milano (Italy)

tel. +39 02 864871 – fax +39 02 8052886

e-mail hoepli@hoepli.it

www.hoepli.it



Tutti i diritti sono riservati a norma di legge
e a norma delle convenzioni internazionali

Indice

Unità 1

Dal problema al programma



L 1 Dal problema all' algoritmo	2
I problemi e la loro soluzione	2
Un problema con la bilancia	5
Il concetto di algoritmo	7
Algoritmi ed esecutori	8
Verifica... le conoscenze	10
Verifica... le competenze	11
L 2 La codifica degli algoritmi: pseudocodice e flow chart	12
I linguaggi per descrivere l'algoritmo	12
I diagrammi a blocchi o flow chart	15
Realizzare i primi diagrammi a blocchi	16
Le variabili e le costanti	19
Conclusione	21
Verifica... le conoscenze	22
Verifica... le competenze	23
L 3 Realizzare i flow chart con Flowgorithm	24
Premessa	25
Ciao mondo!	25
Codifica in Flowgorithm dell'esercizio descritto nella Lezione 2	28
Verifica... le competenze	34
L 4 L'istruzione di selezione e le condizioni logiche	36
Programmi con percorsi alternativi	36
L'istruzione di selezione doppia	37
La selezione con Flowgorithm	40
La selezione semplice	42
Verifica... le competenze	44
L 5 L'istruzione di iterazione (o ciclo)	46
L'istruzione di iterazione o ciclo	46

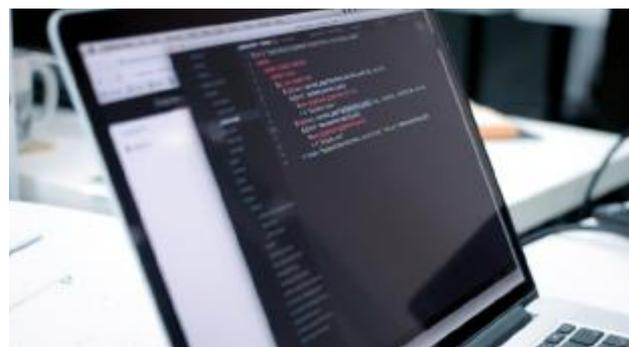
Codificare l'iterazione	47
Iterazione indefinita con Flowgorithm	48
Iterazione definita con Flowgorithm	55
Diagrammi di flusso e programmazione strutturata: il teorema di Böhm e Jacopini	57
Verifica... le competenze	59
Verifica... i saperi essenziali	61
Simulazione guidata di compito in classe	62
Scuola-lavoro	63
CLIL	64

Area digitale

- ⬇ • Origini del problema del contadino, il lupo, la capra e il cavolo
 - Da dove viene il termine algoritmo?
 - Soluzione del problema dell'annaffiatoio
 - La torre di Hanoi e la fine del mondo
 - Esempio di composizione errata delle figure fondamentali
- ✓ • Esercizi interattivi
- ⬇ • Esercizi per l'approfondimento

Unità 2

Programmare in C++ e Java



L 1 I linguaggi per la programmazione degli elaboratori	66
La comunicazione con l'elaboratore	66
I linguaggi di programmazione o ad alto livello	69
Come sono fatti i linguaggi di programmazione?	70
La classificazione dei linguaggi in paradigmi	71
Dal codice sorgente all'esecuzione del programma	72
Compilatori e interpreti	73
Da C a C# fino a Xamarin	76
Da C++ a Java	77
Struttura di un programma OOP	78
Ambienti di sviluppo per C++ e Java	79
Verifica... le conoscenze	80

L 2 Il programma e le variabili

Struttura di un programma	81
I dati e le variabili	81
Assegnare un valore a una variabile	82
Un programma che utilizza i dati	84
Scambiare il contenuto di due variabili	88
Le costanti	88
Verifica... le conoscenze	90
Verifica... le competenze	92

L 3 L'input e l'output dei dati

La comunicazione con l'elaboratore	94
L'input e l'output in C++	94
L'input e l'output in Java	95
Verifica... le competenze	97

L 4 Metodologia per la risoluzione di un problema

Le istruzioni in sequenza	102
Risolvere un problema	103
Un programma per il calcolo di una fattura	104
Verifica... le competenze	107
Verifica... i saperi essenziali	110
Simulazione guidata di compito in classe	111
Scuola-lavoro	112
CLIL	113

Area digitale

- Cronologia dei linguaggi di programmazione
- Tipi e valori per le variabili
- Fare l'input senza la classe Scanner

- Esercizi interattivi

Unità 3

La selezione



L 1 L'istruzione di selezione semplice e doppia

Percorsi alternativi nel programma	116
Verifica... le conoscenze	116
Verifica... le competenze	124

L 2 La selezione annidata e multipla	126
La selezione annidata o nidificata	126
La selezione multipla	133
Verifica... le competenze	138
L 3 Gli operatori logici nella selezione	139
Variabili booleane e proposizioni logiche	139
I connettivi logici	140
Priorità degli operatori	146
Verifica... le conoscenze	148
Verifica... i saperi essenziali	149
Simulazione guidata di compito in classe	150
Scuola-lavoro	151
CLIL	152

Area digitale

- Altri due esempi con l'istruzione switch
- Analogia elettrica degli operatori logici

- Esercizi interattivi

- Esercizi per il recupero

Unità 4

L'iterazione indefinita e definita



L 1 L'istruzione di iterazione precondizionata

Il ciclo a condizione iniziale: while ... {...}	154
La trace table o tabella di traccia	157
Calcolo del massimo comun divisore (MCD) con l'algoritmo di Euclide	161
Verifica... le conoscenze	163
Verifica... le competenze	164

L 2 L'istruzione di iterazione postcondizionata

Il ciclo a condizione finale: do {...} while	165
Contatore e accumulatore	167
Generazione di numeri casuali	169
Verifica... le conoscenze	173
Verifica... le competenze	174

L 3 L'istruzione di iterazione definita

Il ciclo a conteggio	175
Un ciclo dentro un ciclo: i cicli annidati	180
Equivalenza delle istruzioni di iterazione	182
Verifica... le conoscenze	185
Verifica... le competenze	186
Verifica... i saperi essenziali	187
Simulazione guidata di compito in classe	188
Scuola-lavoro	189
CLIL	190

Area digitale

-  • Euclide di Alessandria
-  • Esercizi interattivi
-  • Esercizi per l'apprendimento
 - Esercizi per il recupero
 - Esercizi per l'approfondimento

Unità 5**La scomposizione in sottoprogrammi****L 1 Funzioni e procedure**

Introduzione	192
Sottoprogrammi e funzioni: schema di funzionamento	194
Funzioni in linguaggio C++	195
Funzioni in linguaggio Java	198
Un esempio completo	201
Verifica... le conoscenze	205
Verifica... le competenze	206

L 2 La condivisione delle variabili nei sottoprogrammi

Modello cliente-servitore	207
Ambiente locale e globale	208
Un esempio completo con le variabili globali	209
Struttura di una applicazione software	212
Verifica... le competenze	214

L 3 Le modalità di passaggio dei parametri alle funzioni

Passaggio per valore e per indirizzo	215
Passaggio dei parametri in C++	217
Passaggio dei parametri in Java	217
Un esempio con passaggio per riferimento	218
Conclusioni	219
Verifica... le competenze	220

L 4 Le funzioni ricorsive

Introduzione	221
La ricorsione	222
Schema concettuale della funzione ricorsiva	224
La versione ricorsiva dell'algoritmo di Euclide	227
Da iterazione a ricorsione "tail"	229
Verifica... le competenze	232
Verifica... le competenze	234
Simulazione guidata di compito in classe	236
Scuola-lavoro	237
CLIL	238

Area digitale

-  • Esercizi interattivi

Unità 6**Dati strutturati e algoritmi classici****L 1 Dati strutturati semplici: gli array**

Introduzione ai dati strutturati	240
Il vettore o array monodimensionale	241
I vettori in C++	243
I vettori in Java	244
Utilizzare i vettori	245
Verifica... le competenze	250

L 2 Dati strutturati semplici: le stringhe

Introduzione alle stringhe	251
Le stringhe in C++	252
Le stringhe in Java	258
Verifica... le competenze	262

L 3 La ricerca e la disposizione degli elementi in un vettore

Introduzione	263
Ricerca sequenziale o lineare	264
Il passaggio del tipo array come parametro alle funzioni	267
Analisi dei dati presenti in un vettore	268
Disposizione dei dati in un vettore	270
I vettori paralleli	275
Verifica... le competenze	278

L 4 L'ordinamento degli elementi di un vettore

Introduzione	279
Ordinamento per scambio	280
Bubble-sort parametrico	285
Ordinamento per inserzione	286
Ordinamento per selezione	289
Ordinamento di vettori paralleli	292
Verifica... le competenze	294

⬇ L 5 Due algoritmi evoluti: la ricerca dicotomica e l'ordinamento quicksort

Puoi scaricare la **Lezione 5** anche da  hoeplicuola.it

L 6 Le matrici, array a due dimensioni

Array a due dimensioni	297
Utilizzare le variabili di tipo matrice	297
Definizione di una matrice	299
Utilizzare le matrici rettangolari	301
Un esempio completo: temperature estive	303
Matrice quadrata	306
Verifica... le competenze	310

L 7 I record e le tabelle

Introduzione ai dati strutturati	312
I record e le tabelle in C++	314
I record in Java	316
Verifica... le competenze	320
Verifica... i saperi essenziali	321
Simulazione guidata di compito in classe	322
Scuola-lavoro	323
CLIL	324

Area digitale

- ⬇ • Un esempio con i vettori di caratteri in C++
 - Cosa avviene quando si modifica una stringa
- ⬇ • Esercizi per l'approfondimento
- ⬇ • Proposte di compito in classe

Unità 7

I file



L 1 I file sequenziali

Gli archivi	326
Organizzazione degli archivi	327
Rappresentazione dei dati negli archivi	328
Le operazioni comuni sui file	329
File in C++ e Java	330
Creazione, apertura e chiusura di file di testo in Java	335
Verifica... le competenze	339

L2 I file binari

Tipologia di file binari	341
File ad accesso diretto (random)	343
Scrittura e lettura di record in un file random	346
Ricerca diretta in un file random	350
Verifica... le competenze	353
Verifica... i saperi essenziali	355
Simulazione guidata di compito in classe	356
Scuola-lavoro	357
CLIL	358

Area digitale

- ⬇ • La classe File in Java: i suoi metodi e la sua gerarchia
 - File binari in Java e serializzazione
- ⬇ • Esercizi per il recupero e il rinforzo

Unità 8

La programmazione a oggetti e le interfacce grafiche



L 1 OOP: evoluzione o rivoluzione?	360	Simulazione guidata di compito in classe	442
Introduzione	360	Scuola-lavoro	443
Crisi del software e OOP	362	CLIL	444
Astrazione, oggetti e classi	363		
Conclusione: che cos'è la programmazione a oggetti	364	Area digitale	
Verifica... le conoscenze	367	<ul style="list-style-type: none"> • Leggi catastrofiche • Naming delle classi • Metodi inline e offline in C++ • Installazione di NETBEANS 	
L 2 Classi e oggetti	368	• Esercizi interattivi	
Programmazione modulare	368	• Esercizi per l'approfondimento	
Gli oggetti e le classi	369		
Rappresentazione in UML	370		
Verifica... le conoscenze	375		
Verifica... le competenze	376		
L 3 Metodi e creazioni di oggetti	377		
La scrittura dei metodi	377		
Creazione di oggetti in C++	381		
Creazione di oggetti in Java	383		
Invocazione dei metodi	383		
Un esempio completo	383		
Un esempio con la classe di prova	385		
Verifica... le conoscenze	388		
Verifica... le competenze	389		
L 4 Ereditarietà, polimorfismo e relazioni tra le classi	390		
Generalizzazione ed ereditarietà	390		
Definizioni	393		
Ereditarietà: modalità operative	395		
Realizzazione di una gerarchia	399		
Ereditarietà multipla	404		
Verifica... le competenze	407		
L 5 Ambiente visuale e interfaccia grafica	408		
L'interfaccia utente	408		
Elementi di una interfaccia grafica	410		
L'interfaccia grafica in C#	411		
L'interfaccia grafica in Java	412		
Realizzare il primo progetto visuale in C#	413		
Realizzare il primo progetto visuale in Java	414		
Verifica... le competenze	419		
L 6 I componenti dell'interfaccia grafica	420		
I controlli	420		
Il pulsante di conferma	421		
La casella di controllo	423		
I pulsanti di opzione	425		
I combo box	428		
I list box	430		
Le immagini in C#	433		
Le immagini in Java	435		
Verifica... le competenze	437		
Verifica... i saperi essenziali	439		
		Unità 9	
		Strutture dati dinamiche	
			
		L 1 Le variabili dinamiche in C e C++	446
		Introduzione	446
		Il tipo puntatore	447
		Operazioni con i puntatori	451
		Vettori e puntatori	452
		Composizione di tipi puntatori	454
		Allocazione e deallocazione dinamica di memoria in C++	455
		Verifica... le competenze	459
		L 2 La gestione dinamica della memoria con la classe Vector	461
		Le strutture dati	461
		Gli array dinamici	465
		Un'applicazione: la gestione delle collisioni in una hash table	469
		Verifica... le competenze	472
		L 3 Le liste concatenate in C++	473
		Le liste: strutture di record collegate con puntatori	473
		Implementazione delle liste con puntatori espliciti	474
		Implementazione delle liste con la classe List	482

Le liste bidirezionali e circolari	486
Verifica... le competenze	491

L 4 Le liste concatenate in Java	494
Implementazione della lista in Java	494
La TDA LinkedList per realizzare liste semplici e bidirezionali	497
Verifica... le competenze	501

L 5 Strutture dati ad accesso limitato: pile e code	503
Liste con accesso limitato: pile e code	503
Pila o stack	504
La coda	507
Verifica... le competenze	510

L 6 Gli alberi	511
Introduzione agli alberi e ai grafi	511
Gli alberi binari	515
Visite agli alberi binari	518
Alberi binari di ricerca (ABR)	521
Verifica... le competenze	525

L 7 I grafi	527
Introduzione	527
Definizione e terminologia	529
Rappresentazione dei grafi	531
Visite dei grafi	532
La ricerca del cammino ottimo	533
Grafi euleriani	534
Grafi hamiltoniani e problemi intrattabili	535
Problemi classici	535
Verifica... le competenze	537
Verifica... i saperi essenziali	538
Simulazione guidata di compito in classe	539
Scuola-lavoro	540
CLIL	542

Area digitale

- 
 - Dangling reference
 - La funzione malloc() e free() del linguaggio C
 - Calcolo dei nodi e delle foglie di un albero completo
 - Due applicazioni: la notazione polacca e la codifica di Huffman
 - Alberi in C++ con la classe set<T> della libreria STL
 - Visita in ampiezza e in profondità
 - Soluzione del problema dei 4 colori
- 
 - Esercizi interattivi
- 
 - Esercizi per il recupero e il rinforzo
 - Compiti di realtà per l'approfondimento risolvibili con le liste

Unità 10

Android e i dispositivi mobili



L 1 Android: un sistema operativo per applicazioni mobili	544
I dispositivi mobili e la piattaforma Android	544
La struttura di un'applicazione Android	547
Il ciclo di vita di una Activity	549
Il file APK	550
Verifica... le conoscenze	552

L 2 L'ambiente per lo sviluppo di applicazioni Android	553
Android Studio	353
Creare un'applicazione	554
L'ambiente di lavoro	557
Il Project Explorer	558
Il collaudo della applicazione mediante un emulatore	559
Configurazione di un dispositivo fisico	562
Mandare in esecuzione una app	563
Effettuare il debug con Android Studio	564
Toast	565
Verifica... le conoscenze	567
Verifica... le competenze	567

L 3 Realizzare un'applicazione utilizzando i widget	568
La modifica del layout	568
Widget di base	571
Altri widget molto utilizzati	574
Verifica... le competenze	576
Verifica... i saperi essenziali	577
Simulazione guidata di compito in classe	578
Scuola-lavoro	579
CLIL	580

 L 4 Utilizzare i sensori nella app	
---	--

 L 5 Realizzare una app completa: la calcolatrice	
---	--

Puoi scaricare le **Lezioni 4, 5** anche da  hoeplicola.it

Area digitale

- 
 - Versioni di Android
 - I diversi tipi di tocco su display touch
 - Scaricare e installare Android Studio
 - Utilizzo dei listener
 - Il layout degli elementi grafici
 - Esempio riepilogativo: riassumi la mia identità
- 
 - Esercizi interattivi
- 
 - Esercizi per il recupero e il rinforzo

Unità II

HTML, CSS e JavaScript



L 1 Il linguaggio HTML

Siti Web statici e dinamici	582
HTML	583
La sintassi HTML	584
Il corpo del documento	586
La formattazione del testo	587
Le liste numerate e puntate	590
L'inserimento di immagini	593
I link	594
Le tabelle HTML	596
Inserire file audio e video	599
Verifica... le conoscenze	601
Verifica... le competenze	602

L 2 I fogli di stile (CSS)

I CSS	604
Gli stili	605
L'applicazione degli stili	605
Definizione di regole per più selettori	606
L'applicazione degli stili in cascata	608
Classi e pseudoclassi	609
Le strategie di layout	609
Verifica... le conoscenze	614
Verifica... le competenze	615

L 3 I form e HTML 5

617

HTML 5	618
I form	618
Verifica... le competenze	623

L 4 JavaScript

JavaScript	625
Gli oggetti riflessi del browser	626
La manipolazione degli oggetti del browser	627
La convalida dei moduli	628
Verifica... le conoscenze	628
Verifica... le competenze	632
Verifica... le competenze	632
Verifica... le competenze	634
Simulazione guidata di compito in classe	635
Scuola-lavoro	636
CLIL	638

Area digitale

- 
 - Tipologie di siti Web
 - Link a punti interni
 - Riassunto TAG HTML
 - Script da file esterni
 - Il debugger di Internet Explorer per JavaScript
- 
 - Esercizi interattivi
- 
 - Esercizi per il recupero e il rinforzo

Unità I2

Analisi della complessità computazionale

639



L 1 Elementi di informatica teorica

L 2 La qualità degli algoritmi: introduzione alla complessità computazionale

L 3 La complessità dei problemi

Puoi scaricare l'Unità 12 anche da  hoepliscuola.it

Presentazione

Informatica in C++ e Java tratta la programmazione imperativa e a oggetti per il secondo biennio del corso di Informatica per gli **Istituti Tecnici Tecnologici**, articolazione Informatica e Telecomunicazioni.

La novità dell'opera, concepita secondo le recenti indicazioni ministeriali, tiene conto delle indicazioni ricevute dai docenti che hanno in uso la precedente edizione: in particolare il percorso del **secondo biennio** è stato organizzato in **un volume unico** in modo da favorire l'**autonomia** dei docenti **nell'organizzazione dei percorsi didattici** per il terzo e quarto anno di corso.

In aggiunta, oltre al nuovo impianto grafico, la proposta didattica fornisce le basi teoriche e pratiche della programmazione imperativa e della programmazione a oggetti proponendo parallelamente le codifiche in due diversi linguaggi di programmazione: **C++ e Java**.

STRUTTURA DEL TESTO, METODOLOGIA E STRUMENTI DIDATTICI

Il volume si articola idealmente in **otto sezioni** strutturate in **12 Unità di apprendimento** suddivise in brevi **Lezioni** presentate mediante una **mappa concettuale** funzionale per una **didattica inclusiva**, che riguardano i seguenti argomenti:

- dal problema all'algoritmo;
- la codifica degli algoritmi in linguaggio C/C++ e Java, con particolare attenzione agli algoritmi di ordinamento e ricerca;
- le strutture di dati dinamiche;
- la programmazione a oggetti in C++ e Java;
- la programmazione del Web, con un'unità didattica dedicata a HTML (con riferimenti a HTML 5) e CSS ed esempi di progettazione del layout web;
- la programmazione del Web, con due Unità dedicate a Javascript e alle sue applicazioni;
- la programmazione di app per dispositivi mobili mediante l'utilizzo di Android Studio;
- lo studio della complessità di calcolo degli algoritmi.

Tutti gli algoritmi sono presentati mediante la progettazione top-down, riportando più affinamenti successivi fino a giungere alla codifica in pseudolinguaggio e alla sua rappresentazione con flow chart: la codifica espressa in più di un linguaggio di programmazione, cioè proponendo in parallelo il linguaggio **C++/C# e Java**, si pone come obiettivo di far sviluppare nello studente un pensiero critico di analisi, secondo la teoria del **pensiero computazionale**.

Le reti di computer e il significato di Internet e del Web sono l'argomento che introduce il linguaggio HTML. Vengono presentati i fogli di stile e HTML 5 per definire il layout dei siti Web. La programmazione di app per dispositivi mobili viene esposta mediante l'utilizzo di **Android Studio**.

Ogni Unità si conclude con una **nuova** sezione per la **preparazione al compito in classe** e una simulazione operativa delle attività previste dai PCTO, Percorsi per le Competenze Trasversali e per l'Orientamento (ex Alternanza Scuola-Lavoro), oltre a una scheda CLIL.

Le finalità e i contenuti dei diversi argomenti affrontati sono descritti dagli obiettivi generali e dalle indicazioni *In questa lezione impareremo*; alla fine di ogni lezione per lo studente sono presenti **esercizi, anche interattivi**, di valutazione delle conoscenze e delle competenze raggiunte suddivisi in domande a risposta multipla, a completamento, esercizi con procedure guidate.

L'opera è funzionale a una **didattica inclusiva** grazie alla presenza di **mappe concettuali** e di **esercitazioni e verifiche per alunni DSA**.

ESPANSIONI DIGITALI

La nuova edizione Openschool consente di:

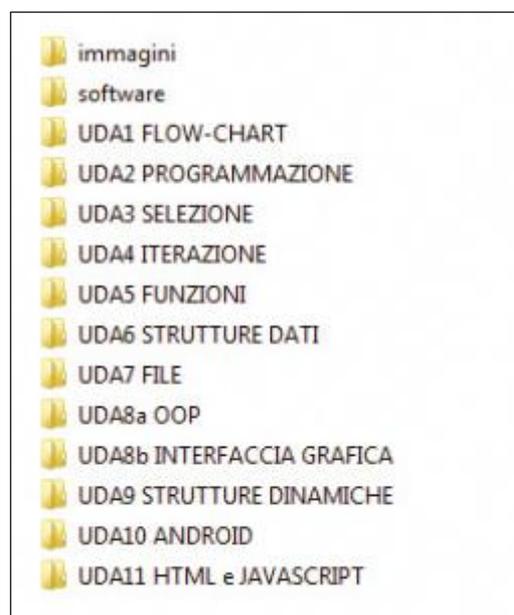
- scaricare gratuitamente il libro digitale arricchito (eBook+); l'eBook+ permette in particolare di:
 - eseguire tutte le esercitazioni a risposta chiusa in modo interattivo;
 - scaricare gli approfondimenti tematici;
 - scaricare lezioni e unità integrative;
- disporre di ulteriori esercitazioni online utilizzabili a discrezione del docente per classi virtuali gestibili attraverso la piattaforma Open.

RISORSE ONLINE E PIATTAFORMA DIDATTICA

Sul sito www.hoepliscuola.it ( hoepliscuola.it) sono disponibili **numerosi materiali**. In particolare, per lo studente: approfondimenti, esercizi di recupero, rinforzo e approfondimento. Inoltre, i **file** richiamati nelle lezioni e nelle esercitazioni contenuti nel CD-ROM allegato al volume sono scaricabili anche dal sito.

CD-ROM

Il CD-ROM allegato al volume contiene i file degli esempi nonché il materiale necessario per eseguire le procedure guidate passo passo degli esercizi svolti e da svolgere e le simulazioni informatiche di fine lezione.



Struttura del corso per immagini



APERTURA UNITÀ

L'Unità si apre con l'indice delle lezioni sviluppate e l'indicazione degli obiettivi generali suddivisi in conoscenze, competenze e abilità.

APERTURA LEZIONE

La Lezione si apre con una breve sintesi degli argomenti trattati e con la schematizzazione dei contenuti attraverso una MAPPA CONCETTUALE.



EVIDENZIAMENTO

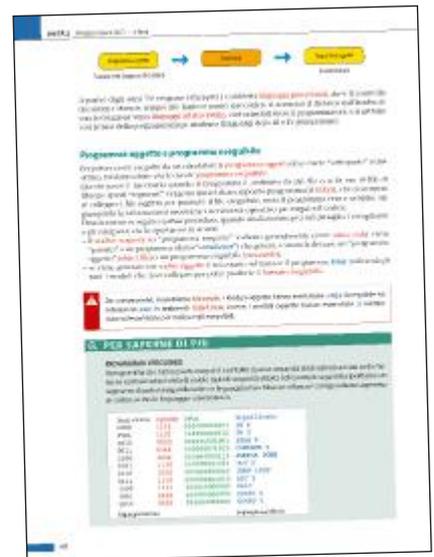
Connota dei concetti da ricordare.



Specifica il significato di un termine.

ATTENZIONE

Individua aspetti su cui focalizzare l'attenzione.

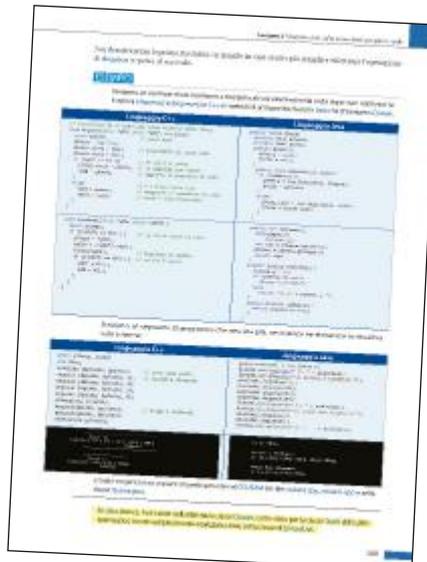


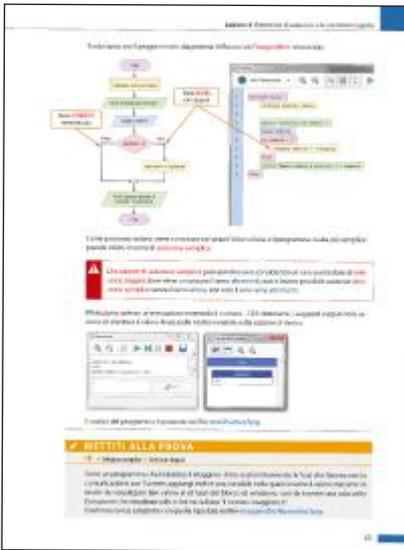
PER SAPERNE DI PIÙ

Schede di approfondimento degli argomenti sviluppati nel volume.

ESEMPIO

Gli esempi chiariscono i concetti appena esposti e svolgono la funzione di traccia di svolgimento per lo studente.





METTITI ALLA PROVA

Appendice esercitativa che prende spunto dal problema di partenza accrescendone le funzionalità e il campo applicativo.



VERIFICA LE CONOSCENZE/COMPETENZE

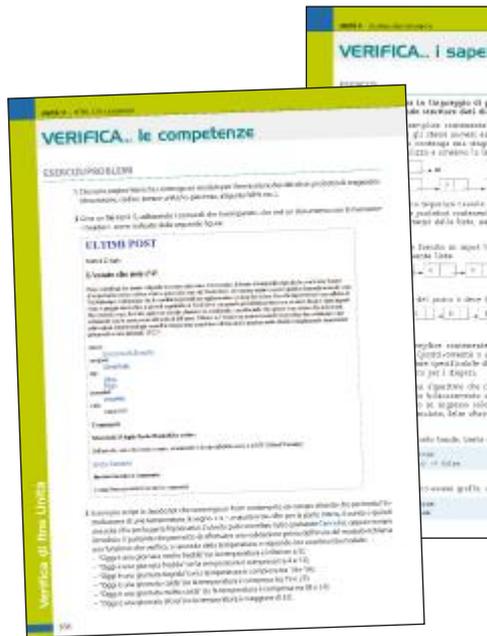
Presenti a fine Lezione, propongono varie tipologie di test (quelli a risposta chiusa sono interattivi e autocorrettivi) e di esercizi, con l'indicazione dei problemi connessi ai compiti di realtà.



VERIFICA I SAPERI ESSENZIALI/ VERIFICA LE COMPETENZE

Al termine di ogni Unità è presente una sezione che contiene esercizi sommativi suddivisi in livelli di difficoltà per la verifica dei saperi acquisiti, con indicati i problemi connessi ai compiti di realtà. **SIMULAZIONE GUIDATA DI COMPITO IN CLASSE**

Presente alla fine di ogni Unità, consente di testare la preparazione prima della verifica in classe.



SCUOLA-LAVORO

Presente al termine di ogni Unità, contiene proposte di attività connesse con le esperienze di PCTO, Percorsi per le Competenze Trasversali e per l'Orientamento.



CLIL

Chiude ogni Unità una scheda CLIL, che propone, in inglese, i concetti chiave dell'Unità e alcuni quesiti di diverse tipologie.



L'OFFERTA DIDATTICA HOEPLI

L'edizione **Openschool** Hoepli offre a docenti e studenti tutte le potenzialità di Openschool Network (ON), il nuovo sistema integrato di contenuti e servizi per l'apprendimento.

Edizione **OPENSCHOOL**



LIBRO DI TESTO



Il libro di testo è l'**elemento cardine** dell'offerta formativa, uno strumento didattico **agile e completo**, utilizzabile **autonomamente** o in combinazione con il ricco **corredo digitale** offline e online. Secondo le più recenti indicazioni ministeriali, volume cartaceo e apparati digitali **sono integrati in un unico percorso didattico**. Le espansioni accessibili attraverso l'eBook+ e i materiali integrativi disponibili nel sito dell'editore sono puntualmente richiamati nel testo tramite apposite icone.

eBOOK+



L'**eBook+** è la versione digitale e interattiva del libro di testo, utilizzabile su **tablet, LIM e computer**. Aiuta a comprendere e ad approfondire i contenuti, rendendo l'apprendimento più attivo e coinvolgente. Consente di leggere, annotare, sottolineare, effettuare ricerche e accedere direttamente alle numerose **risorse digitali integrative**.
→ Scaricare l'eBook+ è molto **semplice**. È sufficiente seguire le istruzioni riportate nell'ultima pagina di questo volume.

RISORSE ONLINE



Il sito della casa editrice offre una ricca dotazione di **risorse digitali** per l'approfondimento e l'aggiornamento. Nella pagina web dedicata al testo è disponibile **MyBookBox**, il contenitore virtuale che raccoglie i materiali integrativi che accompagnano l'opera.
→ Per accedere ai materiali è sufficiente registrarsi al sito **www.hoepliscuola.it** e inserire il codice coupon che si trova nella terza pagina di copertina. **Per il docente** nel sito sono previste ulteriori risorse didattiche dedicate.

PIATTAFORMA DIDATTICA



La **piattaforma didattica** è un ambiente digitale che può essere utilizzato in modo duttile, a misura delle esigenze della classe e degli studenti. Permette in particolare di **condividere contenuti** ed **esercizi** e di partecipare a **classi virtuali**. Ogni attività svolta viene salvata sul **cloud** e rimane sempre disponibile e aggiornata. La piattaforma consente inoltre di consultare la versione online degli eBook+ presenti nella propria libreria.
→ È possibile accedere alla piattaforma attraverso il sito **www.hoepliscuola.it**.

Dal problema al programma

UNITÀ 1



LEZIONE 1
Dal problema all'algoritmo

LEZIONE 2
La codifica degli algoritmi:
pseudocodice e flow chart

LEZIONE 3
Realizzare i flow chart
con Flowgorithm

LEZIONE 4
L'istruzione di selezione
e le condizioni logiche

LEZIONE 5
L'istruzione di iterazione
(o ciclo)

CONOSCENZE

- Conoscere la simbologia dei diagrammi di flusso
- Conoscere la rappresentazione delle figure strutturali
- Acquisire la definizione e le caratteristiche di un algoritmo
- Comprendere la relazione tra algoritmo e programma
- Acquisire il concetto di linguaggio di progetto e di pseudocodifica

COMPETENZE

- Descrivere la soluzione di semplici problemi mediante algoritmi
- Utilizzare le tre figure fondamentali della programmazione
- Acquisire il concetto di variabile e di cella di memoria
- Utilizzare i diagrammi di flusso per rappresentare gli algoritmi
- Codificare i diagrammi di flusso con AlgoBuild

ABILITÀ

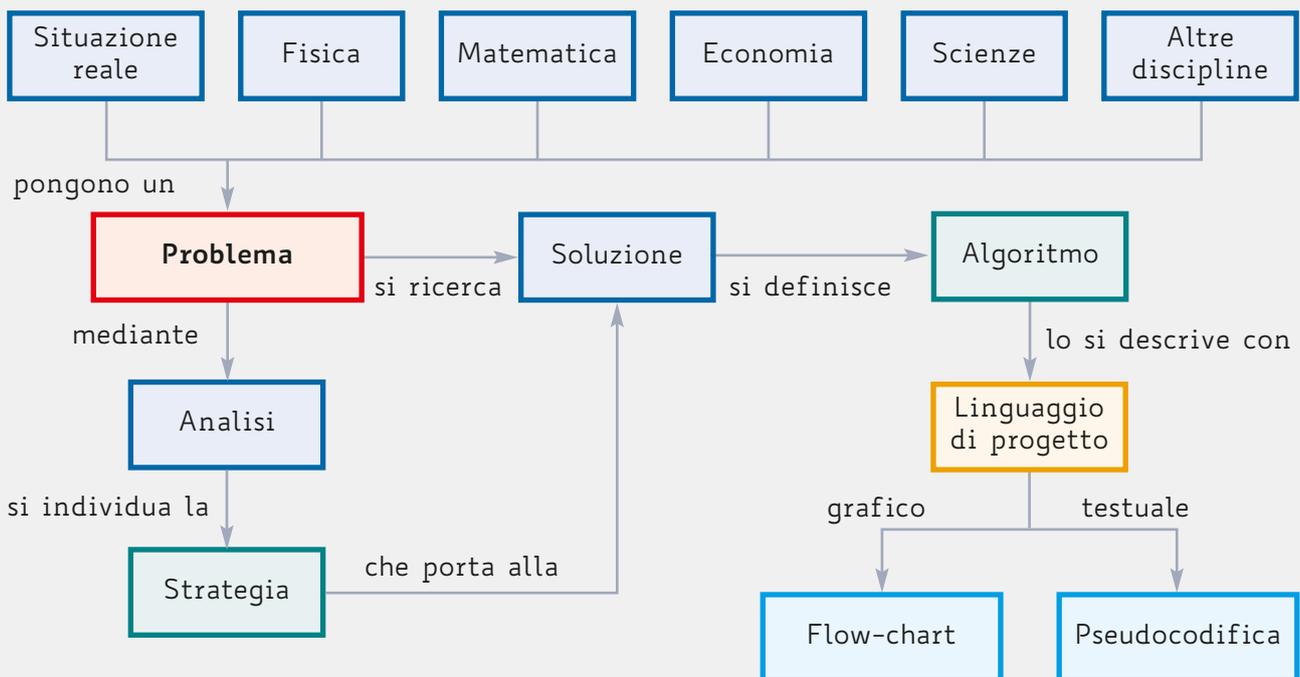
- Formare il pensiero computazionale
- Affrontare in modo sistemico il problema
- Descrivere procedure mediante algoritmi
- Rappresentare gli algoritmi mediante flow-chart
- Memorizzare le informazioni nelle variabili

1 Dal problema all'algoritmo

IN QUESTA LEZIONE IMPAREREMO...

- a riconoscere un problema e a saperlo affrontare
- a riconoscere un algoritmo e un programma

MAPPA CONCETTUALE



Problema

Il matematico e l'informatico identificano con la parola **problema** una **questione** (o quesito) **che deve essere risolta**, della quale viene data una situazione con dei **dati iniziali** noti e un **obiettivo**, che consiste nella soluzione desiderata. La **soluzione** di un **problema** consiste nella definizione della procedura, ovvero delle **operazioni** che devono essere eseguite per **raggiungere lo scopo desiderato**.

I problemi e la loro soluzione

In ogni ambito e settore, dalle scienze all'economia, dalla biologia alla tecnologia, dallo sport alla vita quotidiana, ciascuno di noi deve giornalmente affrontare dei **problemi**, eseguire dei compiti e prendere delle **decisioni**.

ESEMPIO

Ci capita regolarmente di "ripartire il costo" di una pizzata tra amici, di scegliere un film o un regalo da fare a un amico per un compleanno, di dover organizzare la serata, oppure tagliare l'erba in giardino, o mettere in ordine i nostri libri e gli appunti scolastici.

Siamo in grado di risolvere la maggior parte dei **problemi** che affrontiamo perché o sono **semplici**, oppure perché li abbiamo già "affrontati almeno una volta" in passato, e quindi ci basiamo sulla nostra esperienza o sulla consulenza di qualche amico.

Per altri problemi, invece, la ricerca della **soluzione a volte** ci risulta **difficile**, se non impossibile, soprattutto se ci mancano delle conoscenze o delle informazioni (dati), oppure se la difficoltà intrinseca della situazione lo rende un enigma.

ESEMPIO

Abbiamo sicuramente delle difficoltà per calcolare la distanza tra la Terra e il Sole, oppure per mettere in ordine alfabetico i nomi degli abitanti di una città come Pechino o New York, oppure semplicemente per determinare se "è nato prima l'uovo o la gallina!"

I problemi non sono tutti uguali tra loro, né per tipologia, né per complessità.

Per individuare la **soluzione** potrebbe essere necessario cercare di comprendere meglio il problema, cioè effettuare un'**analisi** approfondita della situazione e, nei casi più complessi, individuare una **strategia risolutiva**.



Con **strategia risolutiva** si intende la modalità con cui si risolve il **problema**, cioè l'**idea** con la quale il programmatore, sfruttando l'esperienza, l'intuito, la fantasia e, perché no, l'intelligenza, affronta il processo creativo relativo e trova la "**chiave di soluzione**" del problema.

AREA DIGITALE



Origini del problema del contadino, il lupo, la capra e il cavolo

Per meglio comprendere cosa si intende per **soluzione** e per **strategia risolutiva** riportiamo di seguito un famoso problema, noto come *Il contadino, il lupo, la capra e il cavolo*, con la relativa soluzione.

Il problema

Sulla riva di un fiume ci sono un contadino, un **lupo**, una **capra** e un **cavolo** che devono attraversare un fiume con una piccola barca: su di essa è possibile portare solo due "cose" alla volta.

Come può il contadino attraversare indenne il fiume salvando "capra e cavoli" sapendo che se vengono lasciati da soli il lupo con la capra, oppure la capra con il cavolo, i primi divorano i secondi?



Analisi della situazione

La situazione che ci viene proposta è sufficientemente chiara e definita e non necessita di ulteriori delucidazioni.

Sinteticamente, analizziamo la nostra situazione e riformuliamo il problema:

- abbiamo un **obiettivo** (goal), che è quello di traghettare un contadino e le "tre cose" che ha con sé tra due rive di un fiume;
- abbiamo un **limite** di cose che possiamo portare sulla barca (**vincolo**): due alla volta;
- dobbiamo stare attenti a cosa lasciamo incustodito sulla riva mentre il contadino sta remando (vincolo) in quanto:
 - se rimane il **lupo** con la **capra**... quest'ultima viene mangiata;
 - se rimane la **capra** con il **cavolo**... quest'ultimo viene mangiato.

La definizione della strategia risolutiva

Il nostro **compito** è quello di **trovare** la **soluzione** di questo **problema**, che consiste nel “suggerire” al contadino “**cosa deve fare**”, cioè che tipo di operazione deve eseguire (chi portare sulla barca) e “**come deve farla**”, vale a dire in che ordine. In altre parole, dobbiamo dargli le “istruzioni risolutive” indicandogli la **sequenza** esatta.



La **soluzione** è la tecnica (o l'idea) che consente di risolvere il problema e nell'**informatica** prende il nome di **algoritmo risolutivo**.

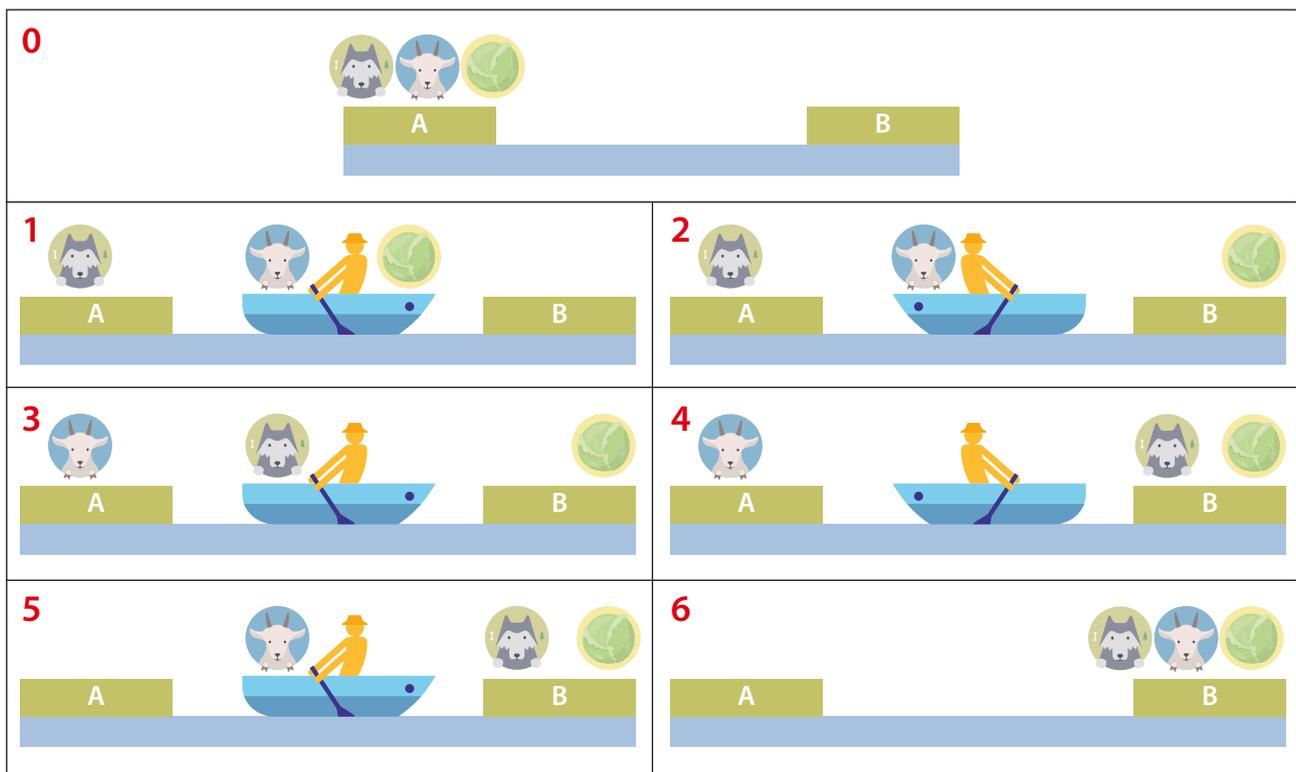
Naturalmente, per trovare la soluzione, dobbiamo riflettere e arrivare alla conclusione applicando le regole prima analizzate.

Dopo qualche tentativo, individuiamo una possibile sequenza corretta delle operazioni:

- 1) prendiamo **capra** e **cavolo** dalla riva A e passiamo sulla riva B;
- 2) lasciamo il **cavolo** sulla riva B e riportiamo la **capra** sulla riva A;
- 3) lasciamo la **capra** sulla riva A, carichiamo il **lupo** sulla barca e lo portiamo sulla riva B;
- 4) torniamo a prendere la **capra** stando “tranquilli” perché il **lupo** “non mangia il **cavolo!**”;
- 5) carichiamo la **capra** e la trasportiamo sulla riva B;
- 6) proseguiamo il nostro viaggio avendo salvato **capra** e **cavolo**.

La soluzione descritta per il problema precedente non è l'unica possibile: si potrebbe infatti sostituire, nella sequenza precedente, il **cavolo** con il **lupo** e il procedimento porterebbe al medesimo risultato: quindi, generalmente, **non c'è una sola soluzione che risolve un problema, ma un problema ammette sempre molte possibili soluzioni alternative**, alcune migliori e più efficienti di altre, ma comunque in grado di soddisfare alle richieste.

È anche possibile descrivere graficamente le operazioni che consentono di risolvere il problema.





Linguaggio naturale

La formulazione di un algoritmo deve essere fatta in modo che sia comprensibile dall'esecutore: se si tratta di esseri umani viene utilizzata la **lingua parlata**, cioè il **linguaggio naturale**.

Esecutore

L'**esecutore** è il soggetto che **esegue l'algoritmo**, che può essere un essere umano oppure una macchina, come un calcolatore elettronico o un automa meccanico.

Al contadino possiamo quindi dare la soluzione sia sotto forma di **istruzioni** scritte in **linguaggio naturale** sia in **formato grafico**: abbiamo **più modalità di rappresentazione della soluzione** e dobbiamo scegliere noi il formato più idoneo al nostro interlocutore: "chi deve eseguire" le istruzioni (**esecutore**) deve essere in grado di comprenderle.

La soluzione viene descritta solitamente indicando l'insieme delle operazioni (**passi**) che devono essere effettuate riportandole nell'ordine in cui devono essere eseguite (**sequenza di operazioni**).

La scelta fatta nell'esempio precedente di indicare la soluzione mediante disegni è sicuramente efficace, in quanto svincola da tante particolari situazioni che potrebbero creare difficoltà, per esempio se il nostro contadino fosse analfabeta oppure di lingua straniera e noi avessimo scritto testualmente i singoli passaggi in italiano.

Gli **esecutori** devono essere in grado di **comprendere ed eseguire le istruzioni** che risolvono il problema.

Un problema con la bilancia

Per illustrare ancor meglio il concetto di **strategia risolutiva**, riportiamo un secondo famoso problema e relativa soluzione, noto come *L'orafo imbroglione e la moneta più leggera*.

Il problema

A un orafo viene consegnato un certo quantitativo d'oro per preparare 18 monete uguali; ma l'orafo "trafuga" parte dell'oro, producendo 17 monete uguali e una più leggera. Utilizzando una bilancia a due bracci, com'è possibile individuare la moneta falsa in sole tre pesate?

Analisi della situazione

La situazione è chiaramente definita: un orafo deve produrre 18 monete d'oro e ne produce 17 uguali, mentre nell'ultima sottrae parte dell'oro e quindi una moneta risulta essere più leggera delle altre.



Il nostro compito (obiettivo), cioè il **goal**, è quello di **individuare la moneta più leggera**.

La definizione della strategia risolutiva

La bilancia che abbiamo a disposizione per smascherare l'imbroglione è quella a due bracci, che non è in grado di indicare il valore numerico del peso ma solo:

- se i due bracci sono in equilibrio, il che significa che su di essi è presente la stessa quantità d'oro;
- se un piatto si abbassa più dell'altro, il che significa che in esso è presente più oro, che ci permette di dedurre che nell'altro piatto è presente la moneta "tarocca"!

Il **vincolo** che abbiamo è quello di **effettuare solo tre pesate!**

In questo caso, per trovare la soluzione non è sufficiente indicare in sequenza le operazioni da compiere perché... innanzitutto è **necessario “scoprire” come arrivare alla soluzione**, cioè avere un’idea di come procedere, **quale strategia adottare**.

Una prima idea consiste nel collocare una metà delle monete su un piatto e l’altra metà sull’altro: in questo modo, individuiamo il “gruppo di 9” dove è presente la moneta falsa.

Se ripetiamo la stessa operazione per altre due volte riusciamo a ridurre il gruppo di 9 monete dapprima a 4 e infine a 2, ma... ci manca una pesata per l’ultimo confronto!

Questo “tentativo” ci fornisce comunque un **indizio**: l’unica possibilità che abbiamo **per risolvere l’enigma** è quello di **avere come ultima pesata solo due monete da confrontare tra loro**.

Anche se alla **terza pesata ci arriviamo con tre monete** possiamo comunque **risolvere il problema**: pesando due monete a caso prese dal gruppo di tre, infatti, abbiamo due possibili situazioni:

- bilancia in equilibrio: la terza moneta, quella “non pesata”, è la più leggera;
- bilancia non in equilibrio: nel piatto “più in alto” c’è la moneta più leggera.

Ripercorrendo il **procedimento a ritroso**, otteniamo che:

- l’**ultima pesata** prevede il **confronto tra due monete** scelte tra tre monete per trovare il risultato;
- la **pesata intermedia deve portare** a individuare un gruppo di tre monete comprendente la moneta tarocca: quindi, se alla seconda pesata abbiamo nove monete da confrontare, possiamo successivamente raggiungere il nostro obiettivo, pesando due gruppi di tre monete:
 - se i bracci sono in equilibrio la moneta falsa è nel terzo gruppo;
 - altrimenti è in uno dei due gruppi sulla bilancia;
- la **prima pesata** deve individuare un gruppo composto al massimo di **nove monete**, che sappiamo di poter “analizzare” con le pesate successive: dato che partiamo con 18 monete, basta posizionarne metà sopra ogni braccio e... il gioco è fatto!

ULTIMA PESATA	PESATA INTERMEDIA	PRIMA PESATA
 <p>3 monete (3 gruppi da 1)</p>	 <p>9 monete (3 gruppi da 3)</p>	 <p>18 monete (2 gruppi da 9)</p>

A questo punto, abbiamo individuato la **strategia**:

- 1 suddividiamo le 18 monete in 2 gruppi da 9 monete;
- 2 effettuiamo la **prima pesata** e individuiamo il gruppo comprendente la moneta più leggera;
- 3 effettuiamo la **seconda pesata** prendendo il gruppo con la moneta falsa e dividendolo in tre gruppi da 3 monete: confrontiamo due gruppi presi a caso e otteniamo, come prima, che:
 - la bilancia è sbilanciata, quindi in un piatto è presente la moneta falsa;
 - la bilancia è in equilibrio, quindi la moneta falsa è nel terzo gruppo;

In entrambe le situazioni siamo quindi in grado di ridurre il numero delle monete da confrontare a sole 3 monete!

- 4 effettuiamo la **terza pesata** prendendo due monete a caso tra quelle presenti nel gruppo di tre:
 - se la bilancia è sbilanciata, in un piatto è presente la moneta falsa;
 - se la bilancia è in equilibrio, la moneta falsa è la terza, quella esclusa dalla pesata.

In entrambi i casi abbiamo trovato la soluzione!

Possiamo osservare che la nostra **strategia può essere utilizzata anche con un problema con dati iniziali diversi** (istanza), per esempio con 24 monete:

- **prima pesata**: suddividendo le monete in tre gruppi da 8 individuiamo il gruppo che contiene la moneta tarocca;

- **seconda pesata**: suddividendo il gruppo di 8 così individuato in tre gruppi rispettivamente di 3-3-2 monete individuamo il gruppo di 3 (o di 2) contenente quella falsa;
- **terza pesata**: siamo arrivati nella medesima situazione del primo esempio.

La **strategia** che risolve un problema permette di **risolvere problemi simili**, cioè quelli della stessa "famiglia", che si differenziano solo per il valore dei dati iniziali (**istanze** di un problema).



Possiamo facilmente constatare che con questo metodo e tre pesate al massimo è possibile individuare una moneta tarocca nascosta tra 27 monete.

PRIMA PESATA	PESATA INTERMEDIA	ULTIMA PESATA
 27 monete	 9 monete	 3 monete

Se aggiungiamo una pesata, con un totale di 4 pesate le monete "confrontabili" divengono 81: possiamo dedurre la regola generale formulandola nel modo seguente: **monete = 3^{nr pesate}**

Il concetto di algoritmo

Abbiamo visto che, partendo dal **problema** per arrivare alla **soluzione**, è necessario **analizzare** dettagliatamente la situazione e individuare la **strategia risolutiva**, cioè "trovare" l'**idea** che risolve la situazione.



Quando la **strategia risolutiva** è stata definita, quindi quando si è "scoperto" il criterio risolutivo del problema, bisogna "scrivere" le singole istruzioni che l'esecutore deve compiere, in **sequenza**, una dopo l'altra.



Algoritmo

Il termine **algoritmo** è una "deformazione" del nome del matematico arabo **al-Khwarizmi**, vissuto nel IX secolo d.C., ritenuto l'ideatore del procedimento che consente di effettuare il calcolo della moltiplicazione tra due numeri mediante la disposizione a cifre incolonnate (che è quella che usiamo ancora oggi).



L'insieme delle **operazioni che permettono di risolvere un problema** prende il nome di **algoritmo**.

Si può anche definire più semplicemente un algoritmo come un "metodo di elaborazione da applicare a certi dati iniziali per ottenere dei dati finali o risultati".

ESEMPIO



Anche la **ricetta** per la preparazione della pizza o di un qualunque piatto è a tutti gli effetti un **algoritmo**, dato che **descrive** a partire dagli ingredienti le **operazioni** che, **in sequenza**, devono essere **eseguite passo passo**, per **ottenere** come risultato il **piatto desiderato**.

AREA DIGITALE



Da dove viene il termine algoritmo?



L'**algoritmo** è quindi una **sequenza ordinata** di passi semplici che hanno lo scopo di **portare a termine un compito** a volte **complesso**.

- Affinché un esecutore possa eseguire un **algoritmo**, questo deve avere le seguenti **caratteristiche**:
- **generalità**, cioè risolvere tutti i problemi di una certa classe: per esempio, l'algoritmo che effettua la somma di due numeri deve portare al risultato corretto indipendentemente dal valore dei due numeri;
 - **finitezza**: l'algoritmo deve essere composto da un numero finito di istruzioni ordinate e deve terminare la sua elaborazione in un numero finito di passi in un tempo finito;
 - **realizzabilità**: l'algoritmo deve essere comprensibile e realizzabile da chi lo deve eseguire, cioè deve essere molto dettagliato e composto di istruzioni molto elementari non ulteriormente scomponibili;
 - **non ambiguità**: l'esecutore deve interpretare in modo univoco le istruzioni;
 - **completezza**: significa che devono essere previste tutte le possibilità che possono verificarsi durante l'esecuzione e per ognuna devono essere definite le azioni da svolgere;
 - **riproducibilità**: ogni volta che viene eseguito l'algoritmo con gli stessi dati di partenza si devono ottenere gli stessi risultati (**determinismo**).

ESEMPIO



È possibile scrivere un algoritmo che descrive le operazioni per "trasferire il Colosseo sulla luna", in quanto esiste un esecutore in grado di compiere tutte le operazioni necessarie alla sua realizzazione, e cioè:

- smontare il Colosseo;
- impacchettarlo;
- portarlo sulla luna, anche effettuando più viaggi;
- rimontarlo sulla luna.

Non sappiamo però in quanto tempo porteremo a termine le operazioni, ma possiamo essere certi che finiranno in un tempo finito, anche se magari "occorrerà qualche anno!"

Non possiamo invece scrivere un algoritmo per colorare le nuvole, oppure per compilare la schedina vincente al totocalcio.

Algoritmi ed esecutori

Oltre agli uomini, i principali **esecutori di algoritmi sono le macchine**: una macchina "non è intelligente", ed è capace solo di eseguire istruzioni molto elementari e senza alcuna capacità critica o di ragionamento autonomo, limitandosi a svolgere automaticamente dei compiti ripetitivi, sulla base delle istruzioni ricevute.

L'"istruzione delle macchine" avviene mediante il procedimento che prende il nome di **programmazione**: questa operazione consiste nel "trascrivere" l'algoritmo mediante un linguaggio particolare che la macchina "è in grado di capire" e di inserirlo al suo interno in modo che lo possa eseguire sotto forma di **programma**.



Indichiamo con il termine **programma** l'insieme delle operazioni che vengono inserite nella macchina, che sono la **traduzione dell'algoritmo** (la sua codifica) **in istruzioni semplici che la macchina è in grado di eseguire**.

Non tutte le macchine sono in grado di eseguire programmi, e inoltre ogni macchina è in grado di eseguire solo un limitato gruppo di istruzioni.

ESEMPIO

**Automa**

Un automa è una **macchina capace di svolgere in maniera automatica** delle operazioni particolari più o meno complesse **che portano a un preciso risultato** per il quale è stata programmata.

Il **distributore di caffè** è una **macchina automatica (automa)** in grado di **ripetere un programma** con molteplici varianti.

Elenchiamo le operazioni che deve essere in grado di svolgere:

- acquisire delle monete e contarle;
- selezionare una bevanda;
- confrontare il costo con le monete inserite;
- se le monete inserite superano il costo del prodotto, predisporre e dare il resto;
- acquisire la quantità di zucchero desiderata;
- iniziare l'erogazione e attendere la terminazione;
- segnalare la disponibilità del prodotto al cliente.

Non è però certo in grado di eseguire il semplice calcolo del prodotto di due numeri.

Anche la **calcolatrice** è una **macchina programmata**, che ha al suo interno un insieme di programmi che ricevono dall'utente un insieme di informazioni in merito a:

- tipo di operazione da eseguire;
- numeri (**dati**) da elaborare.

I dati forniti dall'utente alla macchina per essere elaborati si chiamano **dati in ingresso** (o in **input**).

Il programma esegue i calcoli e visualizza i risultati all'utente sul display, "ritornando" **un risultato in uscita**, cioè in **output**.

Quello appena descritto è lo schema generale di funzionamento di un **sistema di elaborazione**.

Praticamente, **tutti i programmi si basano su questo schema**, composto da una fase di **input**, una di **elaborazione** e una di **output**, che viene ripetuto più volte: pensiamo per esempio ai videogiochi dove il giocatore sfida il computer a un solitario con le carte, o a uno "sparatutto", oppure a un gioco di ruolo o di strategia.



A ogni mossa del giocatore il calcolatore elabora una contromossa e la esegue visualizzando il risultato sullo schermo.

METTITI ALLA PROVA

- • Esame di un problema • Strategia risolutiva • Scomposizione in istruzioni/passi

AREA DIGITALE



Soluzione del problema dell'annaffiatoio

Un agricoltore deve diluire un sacco di fertilizzante in 20 litri d'acqua, ma ha a disposizione un annaffiatoio con capacità di 25 litri contenente 16 litri d'acqua e due contenitori rispettivamente da 5 e 3 litri. Come può operare l'agricoltore?

Descrivi a parole le sequenza delle operazioni necessarie a ottenere esattamente 20 litri d'acqua nell'annaffiatoio.

VERIFICA... LE CONOSCENZE

SCELTA MULTIPLA



1 Quale delle seguenti affermazioni è vera?

- a Gli algoritmi sono problemi
- b I programmi sono problemi
- c I problemi sono algoritmi
- d Gli algoritmi sono programmi

2 Quale di queste affermazioni relative ai problemi è vera?

- a Il calcolatore risolve problemi
- b L'analista studia il problema
- c Il programma risolve un problema
- d La soluzione di un problema dipende dai dati

3 Nella fase di analisi si devono eliminare:

(2 risposte)

- a i reali obiettivi del problema
- b le regole da applicare
- c i dettagli inutili
- d i dati espliciti
- e i dati impliciti
- f i dettagli ambigui

4 La comprensione del problema viene agevolata:
(2 risposte)

- a dalla astrazione
- b dalla lingua utilizzata
- c dal linguaggio di programmazione
- d dalla modellazione

5 Quale tra i seguenti non è uno strumento utile alla definizione della strategia?

- a L'utilizzo dell'esperienza passata
- b La scomposizione dei problemi in sottoproblemi
- c L'utilizzo di un personal computer
- d Il procedimento per tentativi

6 Quale tra i seguenti non è una caratteristica dell'algoritmo?

- a Deve essere generale, cioè risolvere un insieme di problemi
- b Opera su dati in ingresso producendo un risultato in uscita
- c Deve avere un numero preciso di istruzioni
- d Il risultato viene prodotto in un tempo finito
- e Deve essere deterministico

VERO/FALSO



1 Gli algoritmi sono solo per i computer.

2 Gli algoritmi sono programmi.

3 I calcolatori eseguono i problemi.

4 Gli uomini possono eseguire algoritmi.



DOMANDE A RISPOSTA APERTA

1 Indica le differenze tra la fase di analisi e quella di definizione della strategia.

2 Cosa si intende per comprensione del problema?

3 Cosa si intende per strategia risolutiva?

4 Da dove deriva il termine algoritmo?

5 Quali sono le caratteristiche di un algoritmo?

6 Fornisci una definizione di programma.

7 Ricerca, utilizzando Internet, la storia degli automi e delle macchine automatiche.

8 Ricerca la storia della prima "truffa informatica" eseguita col "turco giocatore di scacchi".

VERIFICA... le competenze

AREA DIGITALE

ESERCIZI



Esercizi per
l'approfondimento

Per ciascuna situazione individua un possibile algoritmo risolutivo.



1 I secchi d'acqua

Con l'arrivo della primavera il tuo compito è quello di togliere le erbe infestanti dal sentiero ciottolato: invece che estirparle a mano, quest'anno hai comperato un diserbante che deve essere sciolto in 2 litri d'acqua. Hai però a disposizione solo un primo secchio di 4 litri e un secondo di tre litri. Come puoi fare?

2 Il leopardo, la capra, il topo e il mais

Un pastore deve attraversare un fiume portando sull'altra riva un leopardo, una capra, un topo e un sacco di mais. Ha a disposizione una barca a remi con la quale può traghettare un solo oggetto o animale alla volta. Ma, attenzione! Non può lasciare da soli:

- il leopardo e la capra perché il leopardo mangia la capra;
- il leopardo e il topo perché il leopardo mangia il topo;
- il topo e il mais perché il topo mangia il mais;
- la capra e il mais perché la capra mangia il mais.

Quanti viaggi deve fare per portare sull'altra riva il leopardo, la capra, il topo e il mais?

3 I tre mariti gelosi

Tre mariti e le rispettive tre mogli devono attraversare un fiume su una barca che può trasportare al massimo due persone alla volta. Poiché i mariti sono molto gelosi, nessuna donna deve trovarsi mai assieme ad altri uomini se non in presenza del proprio marito. Come faranno le tre coppie ad attraversare il fiume?
(Pacioli, 1500, *De 3 mariti et 3 mogli gelosi*)

4 Fuga da Alcatraz

Tre innocenti ingiustamente condannati a vent'anni di prigionia decidono di evadere calandosi dalla finestra della loro cella, posta a molti metri di altezza. Il primo pesa 195 kg, il suo degno compare pesa 105 kg mentre il più magro pesa 90 kg. Essi dispongono inoltre di un blocco di cemento del peso di 75 kg. Per fuggire devono utilizzare una corda che scorre su una puleggia. Ai due capi della corda sono fissate due robuste ceste, in ciascuna delle quali può stare un uomo o il blocco di cemento.

Praticamente devono calarsi in varie fasi, facendosi da contrappeso l'uno con l'altro ed eventualmente con il blocco di cemento. La differenza dei pesi nelle due ceste non deve superare i 15 kg, altrimenti la discesa è troppo rapida.

(Lemon, 1890, Lewis Carroll, 1899)

5 Due aerei uno contro l'altro

Un aereo parte da Londra con velocità di 700 km/ora diretto a Pechino. Contemporaneamente un aereo parte da Pechino diretto a Londra alla velocità media di 800 km/ora. Le città distano tra loro circa 10.000 km e hanno 8 ore di fuso orario di differenza. Qual è la distanza tra i due aerei un'ora prima che si incrocino?



LA SFIDA

IL GIOCO DEI FIAMMIFERI

N fiammiferi sono disposti su M file ed è possibile togliere quanti fiammiferi si vuole, ma solo da una fila alla volta (non è quindi ammesso togliere contemporaneamente fiammiferi da due file). Perde chi toglie l'ultimo fiammifero. Individua una strategia per vincere sempre.